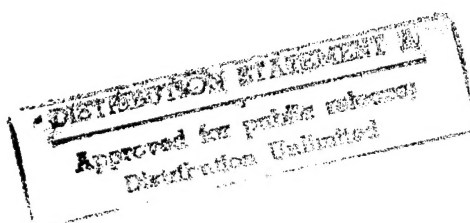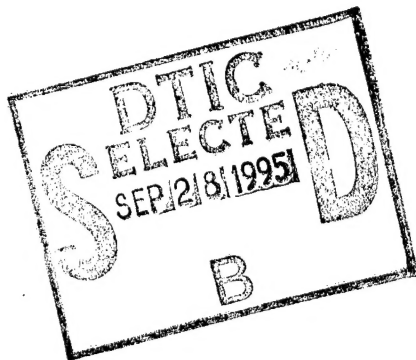A Theory of Distributed Anonymous Mobile Robots
—Formation and Agreement Problems

Ichiro Suzuki
Masafumi Yamashita

TR-94-07-01
July 15, 1994
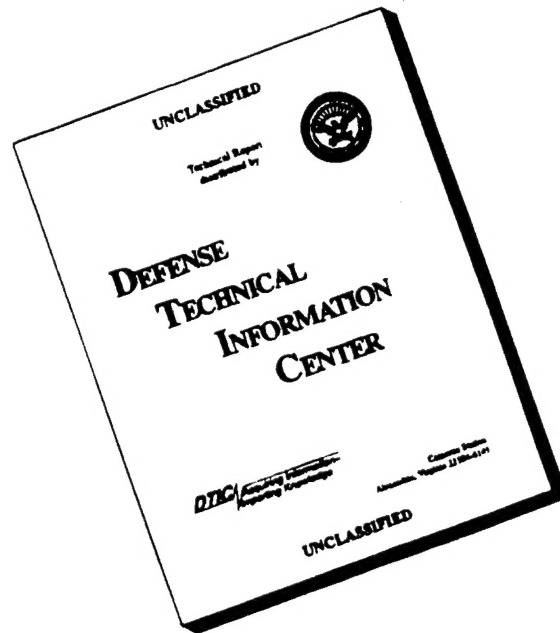
Department of Electrical Engineering and Computer Science
University of Wisconsin – Milwaukee

# DISCLAIMER NOTICE

UNCLASSIFIED

Technical Report
distributed by

**DEFENSE
TECHNICAL
INFORMATION
CENTER**

DTIC

UNCLASSIFIED

THIS DOCUMENT IS BEST
QUALITY AVAILABLE. THE
COPY FURNISHED TO DTIC
CONTAINED A SIGNIFICANT
NUMBER OF PAGES WHICH DO
NOT REPRODUCE LEGIBLY.

19950925 161

Office of Naval Research
Arlington, Virginia

Per Anne Watson, she has no way of getting the

missing pages. The reports that are sent to us is

all that she has. When she get the pages they will

be sent to DTIC with distribution.

February 27, 1996      DSN: 226-4108
                       (703) 696-4108

# A Theory of
# Distributed Anonymous Mobile Robots
# —Formation and Agreement Problems*

Ichiro Suzuki

Department of Electrical Engineering and Computer Science
University of Wisconsin - Milwaukee
P.O. Box 784, Milwaukee, WI 53201, U.S.A.
suzuki@cs.uwm.edu


Masafumi Yamashita
Department of Electrical Engineering
Faculty of Engineering
Hiroshima University
Kagamiyama, Higashi-Hiroshima 724, Japan
mak@se.hiroshima-u.ac.jp

July 15, 1994

**Abstract**     A system consisting of multiple mobile robots in which the robots can see each other by their eye sensors but are not equipped with any communication system, can be viewed as a distributed system in which the components (i.e., robots) can "communicate" with each other only by means of their moves. We use this system to investigate, through a case study of a number of problems on the formation of geometric figures in the plane, the power and limitations of the distributed control method for mobile robots. In the distributed control method, each robot, at infinitely many unpredictable time instants, observes the positions of all the robots and moves to a new position determined by the given algorithm. The robots are anonymous in the sense that they all execute the same algorithm and they cannot be distinguished

by their appearances. The robots are not necessarily synchronous, so they may not always observe their positions simultaneously. Furthermore, initially the robots do not have a common $x$-$y$ coordinate system. The problems we discuss include (1) converging the robots to a single point, (2) moving the robots to a single point, (3) agreement on a single point, (4) agreement on the unit distance, (5) agreement on direction, and (6) leader election. We develop algorithms for solving some of these problems under various conditions. Some impossibility results are also presented.

# 1 Introduction

In the last several years, interest in the distributed control method for multiple mobile robots has increased considerably [1, 2, 7, 10]. The main idea of the method is to let each robot execute a simple algorithm and determine its movement adaptively based on the observed movement of other robots, so that the robots as a whole group will achieve the given goal. This approach has been shown to be very promising for the generation of certain patterns and collision avoidance. In the earlier works on distributed robot control, the main emphasis is on the development of heuristic algorithms for various problems and the presentation of simulation results, and in many cases, formal discussions on the correctness and performance of the algorithms are not given [1, 7].

A robot system in which the robots can communicate with each other by radio, such as a system of radio-controlled vehicles or spaceships, can be considered as a distributed system whose communication topology is a complete graph. Therefore, such systems can be analyzed using the standard techniques developed for distributed computing systems (although such analyses are by no means easy). In this paper, we consider a system consisting of multiple mobile robots in which the robots can see each other by their eye sensors, but they are not equipped with any communication system. The study reveals delicate interplay of a number of key concepts of distributed computing, such as synchrony and asynchrony, communication, termination detection, self-stabilization, anonymity of processors, and knowledge (in a casual sense).

A basic problem for such a robot system is to design an algorithm such that, if all the robots execute it individually, then the robots as a whole group will eventually form the given geometric figure, such as a circle and a line segment [4, 7, 8]. The main goal of this paper is to present some theoretical results related to this problem. The results presented here provide useful insights that will help us to answer certain fundamental questions, such as whether the given algorithm really solves the given problem and, for that matter, whether the given problem is solvable at all in a strict sense by a distributed algorithm. This work is a step toward the ultimate goal of determining exactly what class of problems are solvable in a distributed manner.

We assume that each robot is a mobile processor with infinite memory and an eye sensor, that repeatedly becomes *active* at unpredictable time instants. (At other times it is *inactive*.) Each time a robot becomes active, it observes the positions

2

of all the robots in terms of its own local $x$-$y$ coordinate system, and moves to a new position determined by the given deterministic algorithm.[1] The algorithm is *oblivious* if the new position is determined only from the positions of the robots observed at that time instant. Otherwise, it is not oblivious, and the new position may depend also on the observations made in the past. To simplify the discussion, in this paper we assume that (1) the time it takes for a robot to move to its new position is negligibly small, and (2) a robot is a point (and hence two or more robots can occupy the same position simultaneously). These assumptions help us to bring out the fundamental issues of the problem, and still, many of the techniques and results we obtain for this simplified case seem to apply (with some modifications) to many realistic applications. (We plan to report on the case when the moves of a robot are not instantaneous in a future paper.) The robots are *synchronous* if they always become active simultaneously. Unless otherwise stated, we assume that the robots are not necessarily synchronous. We assume that initially, the robots do not have a common $x$-$y$ coordinate system. So the local $x$-$y$ coordinate systems of two robots may not agree on the location of the origin, the unit distance, or the direction of the positive $x$-axis. The robots are anonymous in the sense that (1) they do not know their identifiers, (2) they all use the same algorithm for determining the next position, and (3) they cannot be distinguished by their appearances. Since a robot observes other robots only at the moments when it becomes active, the third constraint implies that a robot that observes other robots at two time instants may not be able to tell which robot has moved to which position while it was inactive.

Two robots are said to be *clones* of each other if they have the same local $x$-$y$ coordinate system and the same initial position, and they always become active simultaneously. Note that clones can never be separated. Throughout this paper, we assume that no clones exist in the system.

In order to give the reader a concrete image of the robot system, in Section 3 we review some of the known heuristic algorithms for converging the robots to two geometric figures, an approximation of a circle and a line segment. These algorithms are oblivious.

Then we start a formal discussion on the robot system. First, we consider a simple problem of converging the robots toward a single point. (The process of convergence need not terminate in finite steps.) Note that since the robots do not have a common $x$-$y$ coordinate system, we cannot simply use an algorithm such as "move toward point $(0,0)$". For this problem, we give two oblivious algorithms, and then discuss the subtlety of the problem by showing how certain minor changes in the algorithm affect the possibility of achieving the goal. We also consider a related problem of moving the robots to a single point *in finite steps*. Such a problem is called a *formation problem*, in contrast to a convergence problem. We show that this problem can be solved by a nonoblivious algorithm, but not solvable by any oblivious algorithm even for $n = 2$, where $n$ is the number of robots. The corresponding convergence problem

---

[1]Nondeterministic algorithms that allow a robot to randomly select its next position from two or more candidates are out of the scope of this paper.

can be solved by an oblivious algorithm, as we stated above.

Second, we investigate the problem of having the robots agree on a common $x$-$y$ coordinate system. (The term "agree" is defined formally in Section 2.) Clearly, such an agreement can greatly reduce the complexity of motion coordination algorithms. For example, convergence toward a point mentioned in the previous paragraph can easily be solved by moving all the robots toward point $(0,0)$ of the common coordinate system. The problem consists of three subproblems, agreement on the origin, agreement on the unit distance, and agreement on the direction of the positive $x$-axis. We show that the first two agreement problems are solvable by nonoblivious algorithms, but the third problem is not solvable in general, even for $n = 2$. The last result shows that the robots cannot agree on a common $x$-$y$ coordinate system in general.

Third, we consider the case in which the robots have a sense of direction, i.e., the direction of the positive $x$-axis is the same for all robots, and the robots are aware of this fact. For this case, we show that the robots can agree on a common $x$-$y$ coordinate system and elect a leader. We can show that once a unique leader is elected, the robots can be moved to form any geometric figure.

Finally, we consider the case in which the robots are synchronous. It can be shown that in this case, the robots can easily communicate with each other by means of the distances of their moves, once they agree on the unit distance. However, it does not imply that the robots can form any geometric figures, since they may not be able to break certain symmetry in their initial distribution. In fact, whether or not they can form a particular geometric figure depends both on their initial positions and on their local $x$-$y$ coordinate systems. We therefore consider the problem of determining the class of geometric figures that the robots can form starting from the given initial positions and their local $x$-$y$ coordinate systems, using the fact that a synchronous robot system can be viewed as an anonymous complete network, which have been investigated, for example, in [11].

We present necessary definitions and basic assumptions in Section 2. Some of the heuristic algorithms proposed previously are reviewed in Section 3. Convergence and formation problems for a point are discussed in Section 4. Agreement on the origin, unit distance, and direction are discussed in Sections 5, 6 and 7, respectively. In Section 8 we consider the case when the robots have a sense of direction. Section 9 considers the case when the robots are synchronous. Concluding remarks are found in Section 10.

## 2    Definitions and Basic Assumptions

We briefly formalize the problem described in Section 1. Let $r_1, r_2, \ldots, r_n$ be the robots in a two dimensional space. (The subscript "$i$" of $r_i$ is used for convenience of explanation. The robots do not know their identifiers.) We denote by $Z_i$, $1 \leq i \leq n$, the local $x$-$y$ coordinate system of $r_i$. We assume that it is possible that $Z_i \neq Z_j$ for some $i$ and $j$. (If $Z_i \neq Z_j$, then $Z_i$ and $Z_j$ do not agree on one or more of the following: the position of the origin, orientation, and unit distance.) As we will see

4

below, all the positions that $r_i$ observes and computes are given in terms of $Z_i$.

We assume discrete time $0, 1, 2, \ldots$, and let $p_i(t)$ be the position of $r_i$ at time instant $t$, where $p_i(0)$ is the initial position of $r_i$. Define $P(t) = \{p_i(t)|1 \le i \le n\}$ to be the multiset of the positions of the robots at time $t$. ($P(t)$ is a multiset, since we assume that two robots can occupy the same position simultaneously.) For any point $p$, we denote by $[p]_j$ the position of $p$ given in terms of $Z_j$, and define $[P(t)]_j = \{[p_i(t)]_j|1 \le i \le n\}$. Thus $[P(t)]_j$ shows how $r_j$ views the distribution $P(t)$ in terms of its own $Z_j$. Note that if $Z_j \ne Z_k$, then it is possible that $[P(t)]_j \ne [P(t)]_k$, i.e., $r_j$ and $r_k$ may observe distribution $P(t)$ differently. On the other hand, $[P(t)]_j = [P(t)]_k$ may hold even if $p_j(t) \ne p_k(t)$. In this case, $r_j$ and $r_k$ are located at different positions, but $P(t)$ looks identical to them.

The algorithm that a robot uses is a function $\psi$ such that, for any given sequence $(Q_1, p_1), (Q_2, p_2), \ldots, (Q_m, p_m)$ of pairs of a multiset $Q_\ell$ of points and a point $p_\ell \in Q_\ell$, $\psi((Q_1, p_1), (Q_2, p_2), \ldots, (Q_m, p_m))$ is a point. Using $\psi$, we can describe the positions of the robots as follows. At each time instant $t$, each $r_i$ is either *active* or *inactive*. If $r_i$ is inactive at $t$, then $p_i(t+1) = p_i(t)$, i.e., $r_i$ does not move. If $r_i$ is active at $t$, then let $0 \le t_1 \le t_2 \le \cdots \le t_m = t$ be the time instants when $r_i$ was active, and for each $1 \le \ell \le m$, let $Q_\ell = [P(t_\ell)]_i$ and $p_\ell = [p_i(t_\ell)]_i$ be the distribution that $r_i$ observed and the position of $r_i$ at $t_\ell$, respectively. (Note that $Q_\ell$ and $p_\ell$ are given in terms of $Z_i$.) Then $p_i(t+1) = p$, where $p$ is the point such that $[p]_i = \psi((Q_1, p_1), (Q_2, p_2), \ldots, (Q_m, p_m))$. That is, $r_i$ moves to point $\psi((Q_1, p_1), (Q_2, p_2), \ldots, (Q_m, p_m))$ of $Z_i$.

The formalism given above captures the intuition that $r_i$ observes the distribution of the robots only when it is active, and that $r_i$'s next position can depend only on $\psi$ and the distributions that $r_i$ has observed so far. The "$p_\ell$" in pair $(Q_\ell, p_\ell)$ shows that $r_i$ is always aware of its current position in $Z_i$. Algorithm $\psi$ is said to be *oblivious* if $\psi((Q_1, p_1), (Q_2, p_2), \ldots, (Q_m, p_m)) = \psi((Q_m, p_m))$ for any $(Q_1, p_1), (Q_2, p_2), \ldots, (Q_m, p_m)$. In this case, the move of a robot depends only on the current configuration of the robots.

Note that the robots are anonymous in the following sense: (1) function $\psi$ is common to all the robots, (2) the identifier "$i$" of robot $r_i$ is not an argument of $\psi$, and (3) $[P(t)]_i$ contains only the positions of the robots (but not their identities).

The robots are said to be *synchronous* if every robot is active at every time instant; otherwise, they are *asynchronous*. If the robots are asynchronous, then we assume that every robot becomes active at infinitely many unpredictable time instants. In the following, unless otherwise stated we assume that the robots are not necessarily synchronous.

If the robots are asynchronous, then the robots may not be able to obtain a consistent snapshot of their distribution simultaneously. This, as we will see, is a major technical difficulty in designing correct algorithms. For example, if the robots are synchronous, then all the robots observe their initial distribution simultaneously. So they can adopt the centroid (i.e., the arithmetic mean) of their positions as the common origin, and the minimum nonzero distance between any two robots as the common unit distance, assuming that not all robots are located at the same position

initially. So the robots can move to a point on the circumference of the unit circle centered at the origin, and form an approximation of a circle.

Let $\pi$ be a predicate over the set of multisets of points that is invariant under any motion (i.e., rotation and parallel transformation) and uniform scaling. For example, $\pi$ might be true iff the given points are on the circumference of a circle or on a line segment. For such $\pi$, we consider two types of problems, the *convergence problem* and the *formation problem*. In the convergence problem, the goal is to design an algorithm $\psi$ such that, as $t$ goes to infinity, $P(t)$ converges to a distribution that satisfies $\pi$, regardless of the number $n$ of robots, the initial distribution $P(0)$, and (if the robots are not necessarily synchronous) which robots are active at each time instant. The goal of the formation problem is similar, except that the robots must reach some points satisfying $\pi$ in finite steps and "halt". That is, there must exist some time instant $t'$ such that $P(t')$ satisfies $\pi$ and $p_i(t') = p_i(t'+1) = \cdots$ for all $1 \le i \le n$. Since the robots have no knowledge of the underlying coordinate system that we use for describing $\pi$, all we can expect is to have the robots converge to or form a figure similar to the given goal figure. The restriction on $\pi$ stated above was introduced for this reason. All predicates we discuss in the following satisfy this condition.

In addition to convergence and formation problems for a predicate, we discuss *agreement problems* for a given concept $C$, where $C$ might be a location, length or direction. Unfortunately, it is not very convenient to do so within the framework introduced above, since the only property of the robots that are directly observable to us is their movement. So we extend the framework minimally as follows. We imagine that each robot $r_i$ has a *local variable* $\alpha_i$, whose value is *undefined* at time 0. The problem is to design a deterministic algorithm $\psi$ that computes the new positions of a robot *and* the values to be assigned to its local variable, such that, in any scenario that arises under $\psi$, there exists some time instant $t_0$ such that

1. for each $1 \le i \le n$, the value of $\alpha_i$ is defined at $t_0$ and remains unchanged after $t_0$, and

2. the values of $\alpha_1, \alpha_2, \ldots, \alpha_n$ at $t_0$ "agree" on concept $C$.

For example, if $C$ is location, then the second condition requires that each $\alpha_i$ is a position $p_i$ of $Z_i$ and $p_1, p_2, \ldots, p_n$ all refer to the same point $p$, i.e., $[p]_i = p_i, 1 \le i \le n$, for some point $p$. (We can define a similar requirement for agreement for other concepts or combinations of concepts.) This definition of agreement is weaker than that of "common knowledge" that requires "Everyone knows that everyone knows that ... that everyone knows it" for arbitrary depth [3]. It is well-known that if the robots are not synchronous, then they cannot obtain common knowledge that does not exist initially. Leader election is also viewed as an agreement problem, but for this case we require that after some time $t_0$, $\alpha_i = 1$ for exactly one robot $r_i$ (the leader) and $\alpha_j = 0$ for all other robots $r_j$, $j \ne i$.

In this paper, we do not consider dynamic changes in the number of robots while an algorithm is executed. However, using this framework, it is possible to discuss the

situation in which some robots are added and/or removed from the system dynamically. By definition, an oblivious algorithm correctly solves the given problem even if the number of robots changes a finite number of times. For non-oblivious algorithms, we need additional assumptions. Assume that a robot becomes visible (or invisible) when it is added (or removed) from the system, and that if the number of robots changes, then it never changes again until all the robots have noticed the change. Now, modify the given algorithm so that a robot executing it "resets its memory and restarts" (i.e., it ignores the pairs $(Q_\ell, p_\ell)$ for the observations made previously) if it notices a change in the number of robots. Under these assumptions, the modified algorithm correctly works even if the number of robots changes a finite number of times. An algorithm having this property can be viewed as a self-stabilizing algorithm, since it solves the given problem in the presence of transient failures.[2] This is an advantage of the distributed control method. In the centralized method, the entire system can crash if the robot controlling all other robots becomes faulty (and is removed).

# 3  Heuristic Algorithms

Before starting a formal study of algorithms, we first review some known heuristic algorithms for forming a circle and a line segment. These algorithms commonly contain a phase to intersperse the robots more uniformly on a circle or a line segment by moving each robot away from its nearest neighbor, but for simplicity, we omit it in the descriptions given below.

In the following, for robot $r_i$, furthest($r_i$) (resp. nearest($r_i$)) is any of the robots furthest (resp. nearest) from $r_i$ in the current configuration. (Ties can be broken using any deterministic method.) $\nu$ is the maximum distance a robot can move at a time. It is assumed that the robots have a common sense of unit distance.

In the following, when presenting an algorithm, for convenience of discussion we give an informal description of the behavior of a robot or robots executing it, instead of giving a formal definition of function $\psi$. Converting the informal description into a formal definition is straightforward and tedious.

Algorithm $\psi_{circle1}$ given next was proposed by Sugihara and Suzuki [7] for converging the robots to a circle with radius $a$, for given $a$. A detailed investigation of the algorithm is found in Tanaka et al. [9].

**Algorithm $\psi_{circle1}$**
Each time $r_i$ becomes active, it calculates the distance $d$ to furthest($r_i$). If $d - 2a \geq 0$, then it moves distance $\min\{d - 2a, \nu\}$ towards furthest($r_i$). If $d - 2a < 0$, then it moves distance $\min\{2a - d, \nu\}$ away from furthest($r_i$). □

---

[2]A change in the number of robots can be viewed as a transient failure that damages the memory of a robot, except the information on the current distribution of the robots. Since any robot can correctly observe the current distribution, here the term self-stabilizing is used in a weaker sense than the standard one [6].
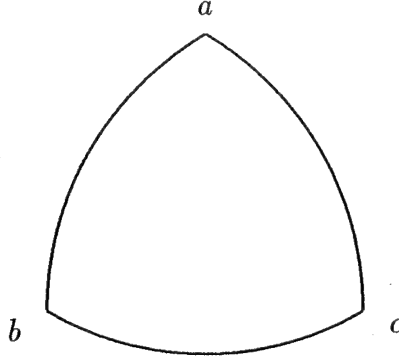
Figure 1: Reuleaux's triangle.

As is pointed out in [7], the robots using $\psi_{circle1}$ sometimes converge to a Reuleaux's triangle (Figure 1), but this small flaw can be remedied considerably by tuning the value of $\nu$ [9]. Further, Tanaka [8] recently proposed a new algorithm $\psi_{circle2}$ given below, and showed, using simulation, that it works better than $\psi_{circle1}$, avoiding convergence to a Reuleaux's triangle when $n$ is large.

**Algorithm $\psi_{circle2}$**
Each time $r_i$ becomes active, it calculates the distance $d$ to the middle point $M$ of nearest($r_i$) and furthest($r_i$). If $d - a \geq 0$, then it moves distance $\min\{d-a, \nu\}$ towards $M$. If $d - a < 0$, then it moves distance $\min\{a - d, \nu\}$ away from $M$. $\square$

As for the problem of forming a line segment, Hirota recently proposed the following algorithm $\psi_{line}$ [4].

**Algorithm $\psi_{line}$**
Each time $r_i$ becomes active, it calculates the distance $d$ to the point $p$ that is the foot of the perpendicular drop from $r_i$ to the line $\ell$ passing through nearest($r_i$) and furthest($r_i$). Then it moves $\min\{d, \nu\}$ towards $p$. $\square$

These algorithms are oblivious and surprisingly simple. This fact seems to demonstrate the usefulness and potential of the distributed approach for controlling the robots. However, the main emphasis of the works mentioned above is on the development of heuristic algorithms and presentation of simulation results, and formal discussions on the correctness and performance of the algorithms are not given. In what follows, we study such distributed algorithms formally. Since the behavior of the robots can be more complex than we might expect, we start the investigation with a very simple problem of converging the robots to a point.

# 4   Convergence and Formation Problems for a Point

We start with two mutually related problems of converging the robots to a point and moving the robots to a point. These problems are perhaps some of the simplest

problems one could consider. Nevertheless, the discussions presented in this section can serve as an introduction to the technical results given in the rest of the paper.

## 4.1 Converging to a Point

Formally, the problem of converging the robots to a point is stated as the convergence problem for predicate $\pi_{point}$, where $\pi_{point}(p_1, \ldots, p_n) = $ true iff $p_i = p_j$ for any $1 \leq i, j \leq n$. We call this problem POINT. We present two oblivious algorithm for POINT and some incorrect variations of one of them.

The following algorithm $\psi_{point1}$ moves each robot $r_i$ toward furthest$(r_i)$, so that the distance between them will be reduced either by a constant factor, or by a constant.

**Algorithm $\psi_{point1}$**
Each time $r_i$ becomes active, it calculates the distance $d$ to furthest$(r_i)$ and moves distance $\min\{d/b_i, \nu_i\}$ towards furthest$(r_i)$. Here, $b_i > 1$ is a constant and $\nu_i > 0$ is the maximum distance $r_i$ can move at a time. ($b_i$ and $\nu_i$ need not be the same for all robots.) □

**Remark 1** Algorithm $\psi_{point1}$ is oblivious. Also, in $\psi_{point1}$, $d$ and $\nu_i$ are given in terms of $Z_i$.

**Theorem 1** *Algorithm $\psi_{point1}$ correctly solves problem POINT.*

**Proof** Recall that $P(t)$ denotes the distribution of the robots at time instant $t$. It suffices to show that $CH(P(t))$ converges to a single point as $t$ goes to infinity, where $CH$ denotes the convex hull. Suppose that $CH(P(t))$ does not converge to a single point. Then since $CH(P(t)) \supseteq CH(P(t+1))$ for any time instant $t$, $CH(P(t))$ must converge to a convex polygon $C$. Let $D$ be the radius of a largest circle that fits in $C$. (If $C$ is a line segment, then let $D$ be one half of the length of $C$.) Since $CH(P(t)) \supseteq C$ for any $t$, the distance between a robot and its furthest neighbor is always at least $D$. So if we choose a constant $\epsilon > 0$ such that $\epsilon < \min_{1 \leq i \leq n}\{D/b_i, \nu_i, D/(1 - 1/b_i)\}$, then when any robot $r_i$ located at point $p$ moves toward its furthest neighbor located at point $q$, the new position of $r_i$ is at distance greater than $\epsilon$ from both $p$ and $q$. Then, since $C$ is a convex polygon, there exists a sufficiently small constant $\delta > 0$ (that depends on $\epsilon$ and $C$) such that, if the new position of $r_i$ is in the $\delta$-neighborhood $\delta_v$ of a corner $v$ of $C$, then at least one of $p$ and $q$ (defined above) is not in the $\delta$-neighborhood $\delta_C$ of $C$. Now, since $\delta$ can be made arbitrarily small, we may choose $\delta$ so that $\delta < \epsilon/2$. Then, since $CH(P(t))$ converges to $C$, there must exist some $t_0$ such that for any $t \geq t_0$, (1) every robot is in $\delta_C$ at $t$ and (2) there is at least one robot in $\delta_v$ of every corner $v$ of $C$ at $t$. However, we can show that the number of robots in $\delta_v$ monotonically decreases after $t_0$, a contradiction to condition (2). To see this, note that for every corner $v$ of $C$, every robot $r_i$ at point $p$ in $\delta_v$ eventually moves, and when it moves, it leaves the $\epsilon$-neighborhood $\epsilon_p$ of $p$. So it leaves $\delta_v$ since $\delta < \epsilon/2$ and $p \in \delta_v$. Suppose $r_i$ enters $\delta_v$ after $t_0$. Then as we mentioned above, either $r_i$ or its furthest neighbor must have been outside $\delta_C$ before the move, a contradiction to condition (1). □

9

Suppose we modify $\psi_{point1}$, so that $b_i = 1$ for all the robots. This means that robot $r_i$ moves to the position of its furthest neighbor, if that neighbor is located within the maximum distance that $r_i$ can move in one step. The modified algorithm does not solve POINT. For example, if (1) there are only two robots and either can move to the position of the other in one move, and (2) they happen to be synchronous, then they will continue to swap their positions forever. Also, $\psi_{point1}$ fails to solve POINT if $d$ is defined to be the distance to a *nearest* neighbor of $r_i$. To see this, consider the case of four robots $r_1, r_2, r_3$ and $r_4$ in which, initially, $r_1$ and $r_2$ are close to each other, $r_3$ and $r_4$ are close to each other, but the $r_1$-$r_2$ pair and the $r_3$-$r_4$ pair are far apart. Suppose that only one robot becomes active at a time. Then, $r_1$ and $r_2$ converge to a point, and $r_3$ and $r_4$ converge to a point, but since this process never terminates in finite steps, the four robots never converge to a point.

Another correct algorithm for POINT is the following. Note that this algorithm is also oblivious.

**Algorithm** $\psi_{point2}$
Each time $r_i$ becomes active, it calculates the distance $d$ to the centroid $g$ of the positions of the robots, and moves distance $\min\{d/b_i, \nu_i\}$ towards $g$. Here, $b_i > 1$ is a constant and $\nu_i > 0$ is the maximum distance $r_i$ can move at a time. ($b_i$ and $\nu_i$ need not be the same for all robots.) □

We can prove the correctness of $\psi_{point2}$ using an argument similar to that in the proof of Theorem 1. The proof is omitted to save space.

## 4.2 Moving to a Point

Next, we discuss the formation problem for predicate $\pi_{point}$ introduced in Subsection 4.1. We call this problem MEET. Since MEET is a formation problem, all the robots must move to a single point *in finite steps*. The next theorem states that problem MEET cannot be solved by any oblivious algorithms, even if the number $n$ of robots is two. Recall that functions $\psi_{point1}$ and $\psi_{point2}$ of Subsection 4.1 are *oblivious* algorithms for solving the corresponding convergence problem POINT.

**Theorem 2** *There is no oblivious algorithm for solving MEET, even for the case* $n = 2$.

**Proof** Suppose that there is an oblivious algorithm $\psi$ that solves MEET for two robots $r_i$ and $r_j$. Note that since $\psi$ is oblivious, the moves of the robots depend only on $Z_i$, $Z_j$ and their current positions. We first show that there exist distinct positions $p$ and $q$ of $r_i$ and $r_j$, respectively, such that either (1) $\psi$ moves $r_i$ from $p$ to $q$, and $r_j$ from $q$ to $q$, or (2) $\psi$ moves $r_i$ from $p$ to $p$, and $r_j$ from $q$ to $p$. (That is, $\psi$ moves exactly one robot to the position of the other, if both robots become active simultaneously.) To see this, assume that such positions do not exist. Consider a scenario $\mathcal{S}$ in which $r_i$ and $r_j$, located at distinct positions $p$ and $q$, respectively, at

10

Figure 2: (a) $r_i$ moves but $r_j$ does not. (b) After modification of $Z_i$.

time $t - 1$, occupy the same position $r$ at time $t$. Now we show that we can modify this scenario and obtain another scenario in which the robots never occupy the same position simultaneously. There are two cases.

**Case 1:** Both $r_i$ and $r_j$ are active at time $t - 1$ in $\mathcal{S}$.
By assumption, $r \neq p$ and $r \neq q$. So if exactly one robot, say $r_i$, happens to be active at $t - 1$, then at time $t$, $r_i$ is located at $r$ and $r_j$ at $q$, where $r \neq q$.

**Case 2:** Exactly one robot is active at $t - 1$ in $\mathcal{S}$.
Suppose that $r_i$ is active at $t - 1$ but $r_j$ is not. Then $r = q$. So if both robots happen to be active at $t - 1$, then at time $t$, $r_i$ is located at $q$ and $r_j$ at some point $s$, where by assumption, $s \neq q$.

Using this argument repeatedly, we can construct an infinite sequence of moves in which the robots never occupy the same position simultaneously. (We can do so in such a way that each robot becomes active infinitely many times, since either of the robots can be chosen to be inactive in Case 1.) So $\psi$ does not solve MEET. This is a contradiction.

So consider an initial distribution $P(0) = \{p, q\}$, where $p \neq q$, in which $r_i$ and $r_j$ are at $p$ and $q$, respectively, and $\psi$ moves $r_i$ from $p$ to $q$, and $r_j$ from $q$ to $q$. See Figure 2(a). (The case in which $\psi$ moves $r_j$ to the positions of $r_i$ is similar.) Now, by modifying $Z_i$ through translation and rotation, we can construct another configuration in which $r_i$ observes distribution $P(0)$ the same way as $r_j$ does, i.e., $[P(0)]_i = [P(0)]_j$ and $[p]_i = [q]_j$. See Figure 2(b). Then $\psi$ moves both $r_i$ and $r_j$ in the same manner in the new configuration, and of course, $\psi$ moves $r_j$ in the same manner in both configurations (namely, from $q$ to $q$). Therefore in the new configuration, $\psi$ moves $r_i$ from $p$ to $p$, and $r_j$ from $q$ to $q$. Then, since $\psi$ is oblivious, both robots remain in their respective initial positions forever. So $\psi$ does not solve MEET. This is a contradiction. $\square$

Theorem 2 shows a limitation of oblivious algorithms. Although oblivious algorithms are easy to understand and analyze, they may not be powerful enough to achieve certain goals. For example, intuitively, the robots' ability to communicate with each other could be severely limited, since memorizing some common concepts (e.g., the size of the unit distance, the direction of north) seems to be inevitable for effective communication.

On the other hand, MEET can be solved for two robots by a *non-oblivious* algorithm. One such algorithm is $\psi_{meet(2)}$ given next. Algorithm $\psi_{meet(2)}$, as well as all other algorithms in the rest of the paper, has been written under the assumption

11

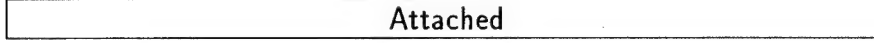Figure 3: Illustration for $\psi_{meet(2)}$.

Figure 4: The case $t_i = t_j$.

that the robots occupy *distinct* positions in the initial distribution. In Appendix A, we give an algorithm for transforming any given distribution of any number of robots (in which some robots may occupy the same position) into one in which (1) no two robots occupy the same position, and (2) the robots are not located on a single line segment if $n > 2$, under the assumption that clones do not exist.

**Algorithm $\psi_{meet(2)}$**
When $r_i$ becomes active for the first time, it translates and rotates its coordinate system[3] $Z_i$ so that

1. $r_i$ is at $(0,0)$ of $Z_i$, and

2. the other robot $r_j$ is on the positive $y$-axis of $Z_i$, say at $(0,a)$ for some $a > 0$.

Then it moves in the positive $x$ direction of $Z_i$, over any nonzero distance. It then continues to move in the same direction each time it becomes active, until it observes that the position of $r_j$ has changed twice. (See Figure 3.)

Now, $r_i$ knows line $\ell$ that contains the first two distinct positions of $r_j$ that $r_i$ has observed. (Note that by symmetry, $\ell$ is the $x$-axis of $r_j$'s coordinate system $Z_j$.) Then using Lemma 1 given below, $r_i$ finds the initial position of $r_j$, and moves to the midpoint of the initial positions of $r_i$ and $r_j$. $\square$

Lemma 1 shows that robots $r_i$ and $r_j$ executing $\psi_{meet(2)}$ eventually find out which of them became active first for the first time, and what their initial distribution was.

**Lemma 1** *Let $t_i$ and $t_j$ be the time instants at which $r_i$ and $r_j$, respectively, become active for the first time in $\psi_{meet(2)}$. Then the following hold.*

1. *The trajectory of $r_i$ and the trajectory of $r_j$ are parallel iff $t_i = t_j$. In this case, each robot sees the other robot at its initial position at $t_i(= t_j)$ (Figure 4).*

---

[3]Formally, $r_i$ cannot modify $Z_i$ in our framework. For convenience of explanation, however, we sometimes imagine that $r_i$ transforms $Z_i$ into a new coordinate system. The effect of such a transformation can easily be simulated within the framework in which $Z_i$ remains unchanged.

Figure 5: The case $t_i < t_j$.

2. *The trajectory of $r_j$ intersects the negative x-axis of $Z_i$ iff $t_i < t_j$. In this case, $r_i$ sees $r_j$ at its initial position, and $r_i$'s initial position is the foot of the perpendicular drop from $r_j$'s initial position to the line containing the trajectory of $r_i$ (Figure 5).*

3. *The trajectory of $r_i$ intersects the negative x-axis of $Z_j$ iff $t_j < t_i$. In this case, $r_j$ sees $r_i$ at its initial position, and $r_j$'s initial position is the foot of the vertical drop from $r_i$'s initial position to the line containing the trajectory of $r_j$.*

**Proof** The lemma follows immediately from the description of $\psi_{meet(2)}$.  □

**Theorem 3** *Algorithm $\psi_{meet(2)}$ solves problem MEET for the case $n = 2$.*

**Proof** The key observation is the following: When $r_i$ observes that the position of $r_j$ has changed twice, $r_j$ must have already observed that $r_i$'s position has changed at least once, and thus $r_j$ knows where the x-axis of $Z_i$ is. Similarly, $r_j$ will know that $r_i$ knows where the x-axis of $Z_j$ is. Then the correctness of $\psi_{meet(2)}$ follows from Lemma 1.  □

Before presenting a non-oblivious algorithm for solving MEET for three or more robots, we discuss a technique that we use often for designing algorithms. One technical difficulty is that in general, a robot cannot determine, given the positions of the robots observed at two time instants $t_1$ and $t_2$, which robot has moved to which position between $t_1$ and $t_2$. One way to overcome this problem is to impose a bound on the maximum distance that any robot can move while other robots remain inactive, so that the robot at position $p$ at time $t_1$ must be at the position closest to $p$ at time $t_2$. Specifically, we do the following. When robot $r_i$ becomes active for the first time, it memorizes the distance $a_i > 0$ to its nearest neighbor. Then $r_i$ moves at most distance $a_i \epsilon / 2^k$ in the $k$-th move, where $0 < \epsilon \leq 1/2$ is a constant chosen by the algorithm. This restriction assures that $r_i$ remains in the interior of the $a_i \epsilon$-neighborhood of its initial position. Since the interiors of such neighborhoods of two robots located at different positions do not intersect, any robot can correctly know the position of $r_i$, even after it remains inactive for a long time. This technique and its variations are used in $\psi_{meet}$ given next, $\psi_{scatter}$ of Appendix A, and some other algorithms.

The following is a non-oblivious algorithm $\psi_{meet}$ for solving MEET for three or more robots. It is assumed that in the initial distribution, (1) no two robots occupy the same position, and (2) the robots are not located on a single line segment. Appendix A explains how a given distribution can be transformed into one satisfying these conditions.

13

Figure 6: Illustration for $\psi_{meet}$.

**Algorithm** $\psi_{meet}$

When $r_i$ becomes active for the first time, it determines whether or not it is located at a corner of the convex hull $C$ of the distribution of the robots at that time instant. There are two cases.

**Case 1:** $r_i$ is not at a corner of $C$.

$r_i$ memorizes the position $p$ of its nearest neighbor and the distance $a_i$ to $p$. Then it moves towards $p$, and continues to move toward $p$ each time it subsequently becomes active, staying in the interior of the $(a_i/2)$-neighborhood of its initial position. (See the explanation given above.)

**Case 2:** $r_i$ is at a corner of $C$.

Let $a$, $b$, $c$ and $d$ be consecutive corners of $C$ in clockwise order, where $r_i$ is at $b$. See Figure 6. Then $r_i$ memorizes the direction that is away from $a$ along the line containing $\overline{ab}$, and moves in that direction each time it becomes active, staying in the interior of the $a_i\epsilon$-neighborhood of $b$ for some $\epsilon \leq 1/2$. Here, $\epsilon$ is chosen so that the $a_i\epsilon$-neighborhood does not intersect the line containing $\overline{cd}$. (This assures that the robot $r_j$ at corner $c$ remains to be at a corner of the convex hulls of the new positions of the robots even after $r_i$ and $r_j$ move.)

Robot $r_i$ continues to move as described above, until it observes that the position of each robot has changed at least twice. Then, $r_i$ knows line $\ell_j$ that contains the first two distinct positions of $r_j$ that $r_i$ has observed, for each robot $r_j$ at a corner of the initial convex hull. Since the convex hull of the initial positions of the robots is a $k$-sided polygon for some $k \geq 3$, the lines $\{\ell_j\}$ determine a unique, smallest convex polygon $Q$ that contains the initial convex hull. Then $r_i$ moves to the centroid of the corners of $Q$. □

**Theorem 4** *Algorithm* $\psi_{meet}$ *solves MEET for* $n > 2$.

**Outline of Proof** Using an argument similar to the one in the proof of Theorem 3, we can show that every robot executing $\psi_{meet}$ eventually knows the lines $\{\ell_j\}$ correctly. (The trajectories of the robots that are not at a corner of the initial convex hull are not used to define $Q$. We move such robots, however, so that other robots $r_i$ will know that they have become active sufficiently many times and observed $r_i$'s movement.) So all the robots eventually know the corners of $Q$. Thus $\psi_{meet}$ solves MEET for $n > 2$. □

By Theorems 3 and 4, MEET is solvable for any $n \geq 2$.

14

# 5  Agreement on the Origin

In this section we discuss the problem of agreement on the origin of a common $x$-$y$ coordinate system. We call this problem ORIGIN. ORIGIN is an agreement problem for concept $C$ as defined in Section 2, where $C$ is a location.

**Theorem 5**  *ORIGIN is solvable for any $n \geq 2$.*

**Proof**  As we have seen, the robots that are initially at distinct positions and executing $\psi_{meet(2)}$ (for the case $n = 2$) or $\psi_{meet}$ (for the case $n > 2$) eventually know the point at which they meet (the midpoint of the initial positions for the case $n = 2$, and the centroid of the corners of $Q$ for the case $n > 2$). So they can agree on that point.  $\square$

# 6  Agreement on the Unit Distance

We call the problem of agreeing on the unit distance UNITDIST. This is an agreement problem for concept $C$, where $C$ is a length. So the goal is to let each robot $r_i$ decide on a length $L_i$ in terms of its own $Z_i$, so that the lengths $L_1, L_2, \ldots, L_n$ chosen by the robots all refer to the same physical length.

**Theorem 6**  *UNITDIST is solvable for any $n \geq 2$.*

**Proof**  For the case $n = 2$, using $\psi_{meet(2)}$, each robot eventually finds the initial position of the other (Lemma 1). Then they can use the distance between their initial positions as the unit distance. For the case $n > 2$, the robots can choose, as the unit distance, the length of a shortest side of the $k$-sided convex polygon $Q$ that they obtained using algorithm $\psi_{meet}$ in Subsection 4.2.  $\square$

The proofs of Theorems 5 and 6 show that the robots can agree on both a point and a length simultaneously. Thus they can agree on a circle whose center is at the agreed point and whose radius is the agreed length. So we have:

**Corollary 1**  *The problem of agreeing on a circle and the problem of forming an approximation of a circle (in the sense that all the robots are located on the circumference of a common circle) are solvable for any $n \geq 2$.*

# 7  Agreement on Direction

Now we show that the third problem of agreeing on direction is unsolvable in general. Let us call this problem DIRECTION. Here, the concept $C$ on which the robots agree is a direction.

15

Figure 7: A symmetric configuration.

**Theorem 7** *There is no algorithm for solving DIRECTION, even if $n = 2$ and the robots are synchronous.*

**Proof** Consider two synchronous robots $r_i$ and $r_j$ such that initially,

1. $r_i$ is located at position $(0,0)$ of $Z_i$ and at position $(0,1)$ of $Z_j$, and

2. $r_j$ is located at position $(0,0)$ of $Z_j$ and at position $(0,1)$ of $Z_i$.

See Figure 7. Then the robots always move in the same (symmetric) manner, and thus when $r_i$ chooses a direction, $r_j$ chooses the opposite direction. ☐

Note that as we stated in Section 2, the other two agreement problems, ORIGIN and UNITDIST, are trivially solvable if the robots are synchronous.

Using an argument, similar to that in the proof of Theorem 7, involving three robots that initially form an equilateral triangle, we can prove the following theorem:

**Theorem 8** *The problem of forming a line segment is unsolvable.* ☐

# 8  Robots with a Sense of Direction

We say that the robots have a sense of direction if they agree on the direction of the positive $x$-axis, which we call "east." (We use "west," "north," and "south" in the understood manner.) The main result of this section is the following.

**Theorem 9** *If the robots have a sense of direction, then they can agree on a common coordinate system, and elect a leader.*

**Proof** The theorem follows from the discussion given below. ☐

Again, it is assumed that initially, (1) no two robots are located at the same position, and (2) the robots are not located on a single line segment if $n > 2$.

The argument for the case $n = 2$ is similar to those in the proofs of Theorems 5 and 6. Using $\psi_{meet(2)}$, robots $r_i$ and $r_j$ can adopt the midpoint of their initial positions as the origin, and the distance between their initial positions as the unit distance. Then, since the robots have a sense of direction, they have agreed on a coordinate system. Finally, the robot whose initial position in the agreed coordinate system is larger in lexicographic ordering can be selected as the leader.

For the case $n > 2$, we present a new algorithm, $\psi_{coordinate}$, that solves *both* the coordinate agreement problem and the leader election problem, when the robots have a sense of direction. (Of course, the possibility of agreement on a coordinate system alone follows immediately from the discussions in Sections 5 and 6.)

16

Figure 8: Algorithm $\psi_{coordinate}$.

**Algorithm** $\psi_{coordinate}$

The idea is to choose (1) the vertical (i.e., north-south) line though the eastern-most robots as the common $y$-axis, (2) the distance between the two vertical lines, one through the eastern-most robots and the other through the western-most robots, as the unit distance, and (3) the horizontal (i.e., east-west) line through the northern-most robots (excluding two special robots, as explained below) as the common $x$-axis. To achieve this, the robots do the following.

1. The northern-most robot among the eastern-most robots continues to move north within a "small" neighborhood of its initial position (as explained in Section 4.2). All other eastern-most robots continue to move west within a "small" neighborhood of its initial position, in such a way that they will never become western-most.

2. The northern-most robot among the western-most robots continues to move north within a "small" neighborhood of its initial position. All other western-most robots continue to move east within a "small" neighborhood of its initial position, in such a way that they will never become eastern-most.

3. The robots that are neither eastern-most nor western-most continue to move west within a "small" neighborhood of its initial position, in such a way that they never become western-most.

See Figure 8. Then eventually, every robot knows the trajectories of all other robots, and chooses the line containing eastern vertical trajectory as the common $y$-axis, and the line containing the northern-most horizontal trajectory as the common $x$-axis. The unit distance is chosen to be the distance between the two vertical trajectories. The leader is the robot moving along the eastern vertical trajectory. $\square$

The correctness of $\psi_{coordinate}$ should be immediate from its description, once we note that, since $n > 2$ and the initial distribution of the robots satisfy the assumptions given above, the two vertical trajectories do not overlap (so the unit distance is well-defined) and there is at least one horizontal trajectory (so the common $x$-axis is well-defined). So Theorem 9 follows.

Since the leader can compute the final positions of all the robots that satisfy any given predicate and "guide" them to their respective final positions, we obtain the following theorem.

**Theorem 10** *If the robots have a sense of direction, then for any predicate $\pi$ that is invariant under any motion and uniform scaling, the convergence and formation problems for $\pi$ are solvable.*

17

**Proof** Let $\pi$ be any given predicate that is invariant under any motion and uniform scaling. We only need to show how the formation problem for $\pi$ can be solved. First, using $\psi_{coordinate}$, the robots elect the leader, say robot $r_i$. Even after $r_i$ is elected, the robots except $r_i$ continue to move within their respective "small" neighborhoods of their initial positions, as specified in $\psi_{coordinate}$. Then $r_i$ computes a final position $p_j$ for every robot $r_j$, such that $p_j$ is to the east of $r_i$'s own vertical trajectory and the multiset $\{p_j\}$ satisfies $\pi$. (Recall that $\pi$ is invariant under any motion. Also, since $r_i$ is one of the eastern-most robots, the final positions $p_j$ are all to the east of the current positions of the robots.) Now, $r_i$ "guides" all other robots to their respective new positions one by one. Specifically, $r_i$ selects one robot $r_j$ at a time and moves to the current position of $r_j$. It is possible that $r_j$ moves to a new position simultaneously, but $r_i$ is now on the trajectory of $r_j$ and within the "small" neighborhood that $r_j$ uses. Since no other robot can enter this region, $r_j$ knows that $r_i$ has selected $r_j$. When $r_i$ observes that $r_j$ moves again, $r_i$ knows that $r_j$ knows that $r_j$ has been selected. So $r_i$ moves to $p_j$ and waits until $r_j$ moves to that point. (Meanwhile, $r_j$ moves to the position where $r_i$ has moved to. Here, if it takes two or more steps for $r_i$ to reach $p_j$, then $r_i$ moves monotonically eastbound to $p_j$, so that $r_j$ knows that $r_j$ has reached $p_j$ when $r_i$ moves westbound after $r_j$ catches up with $r_i$.) Robot $r_i$ does this repeatedly for all other robots, and then moves to its own final position $p_i$. $\square$

In particular, from Theorem 10 we have:

**Corollary 2** *If the robots have a sense of direction, then the problem of forming a line segment is solvable.*

# 9   Synchronous Robots

In this section, we characterize the the class of geometric figures that the robots can form, under the assumption that they are synchronous. Again, we assume that initially, all robots occupy distinct positions. In addition, for simplicity of explanation, we assume that (the robots know that) each robot $r_i$ is located at the origin of its coordinate system $Z_i$ at time 0. Essentially the same results hold even without this assumption.

What geometric figures can be formed depends not only on the given initial positions of the robots, but also on their local $x$-$y$ coordinate systems. For example, suppose that initially, four robots $r_1$, $r_2$, $r_3$ and $r_4$ form a square in counterclockwise order, where $r_2$ is at position $(1,0)$ of $Z_1$, $r_3$ is at position $(1,0)$ of $Z_2$, and so on. See Figure 9. Then the robots have the same "view", and since the clocks are synchronized and the algorithm they use is deterministic, they can never break symmetry (and intuitively, they continue to form a square all the time). On the other hand, if the direction of the positive $x$-axis happens to be the same for all four robots (Figure 10), then the robots can easily discover this fact and elect the robot that is northern-most among the eastern-most robots as the leader (and form any geometric

18

Figure 9: A symmetric configuration of four robots.



Figure 10: A configuration that is not symmetric.

figure). Intuitively, in this case every robot has a unique "view," and thus the robots can elect a leader using a suitable total ordering of the "views." In the following, we formalize this observation.

Following [11], the *view* of robot $r_i$ at time $t$, denoted $V_i(t)$, is defined recursively as a rooted infinite tree as follows.

1. The root of $V_i(t)$ has $n-1$ subtrees, one for each robot $r_j$, $j \neq i$.

2. The edge from the root of $V_i(t)$ to the subtree corresponding to $r_j$ is labelled $((a,b),(c,d))$, where $(a,b)$ is the position of $r_j$ in terms of $Z_i$, and $(c,d)$ is the position of $r_i$ in terms of $Z_j$.

3. The subtree corresponding to $r_j$ is the view $V_j(t)$ of $r_j$ at time $t$.

Note that each vertex of $V_i(t)$ corresponds to a robot, but it is not labelled as such. Two views $V_i(t)$ and $V_j(t')$ are said to be *equivalent*, written $V_i(t) \equiv V_j(t')$, if they are isomorphic to each other including the labels.

$V_i(0)$ is thus the view of $r_i$ when it becomes active for the first time. Note that since the robots occupy distinct positions at time 0, the edges incident on the root of $V_i(0)$ have distinct labels. Since at time 0 the robots have no knowledge of other robots' local coordinate systems, at time 0 robot $r_i$ does not know its view $V_i(0)$.

It is possible for the robots to obtain sufficient information to construct their views. The following algorithm achieves this.

**Algorithm** $\psi_{getview}$
At time 0, the robots adopt the minimum distance between any two robots as the common unit distance. Then each robot $r_i$ moves in the direction of its positive $x$-axis over distance $f(d_i)/2$ measured in terms of the common unit distance, where $d_i > 0$ is the common unit distance measured in terms of $Z_i$ and for $x > 0$, $f(x) = 1 - 1/2^x$ is a monotonically increasing function with range $[0,1)$. When the robots become active at time 1, they all return to their respective initial positions.[4] □

---

[4]If the maximum distance that $r_i$ can move in one step is smaller than $f(d_i)/2$, then we can let $r_i$ continue to move in the same direction until the total distance of the moves is exactly $f(d_i)/2$. Since the robots are synchronous, they can wait until all robots have moved over the desired distances and stop, and then return to their initial positions, possibly using the same technique.

19

At time 1, $r_i$ knows, for each robot $r_j$, the direction of the positive $x$-axis and the unit distance of $Z_j$. Thus, since by assumption the origin of $Z_j$ is the same as the initial position of $r_j$, $r_i$ knows $Z_j$ and the the positions of all the robots in terms of $Z_j$. Using this information, $r_i$ can construct its view $V_i(0)$. Note that since each robot returns to its initial position when it moves at time 1, we have $V_i(0) \equiv V_i(2)$.

Let $m$ be the size of a largest subset of robots having an equivalent view at time 0. If $m = 1$, then every robot has a unique view, and thus once Algorithm $\psi_{getview}$ is executed, using a suitable total ordering over the views, the robot having the largest view can be chosen as the leader, and thus the robots can form any geometric figure, as explained in the proof of Theorem 10. So in the following, we consider the case $m \geq 2$. Lemmas 2, 3, 4 and 5 given below refer to a fixed initial configuration with $m \geq 2$.

**Lemma 2** *The robots can be partitioned into $n/m$ groups of $m$ robots each, such that two robots have an equivalent view iff they belong to the same group.*

**Proof** The claim is trivial if $m = n$. So assume that $m < n$, and without loss of generality suppose that $V_1(0) \equiv V_2(0) \equiv \cdots \equiv V_m(0)$ but $V_1(0) \not\equiv V_{m+1}(0)$. That is, $r_1, r_2, \ldots, r_m$ have an equivalent view at time 0 but $r_{m+1}$ does not. Let $((a,b),(c,d))$ be the label of the edge from the root of $V_1(0)$ to the vertex corresponding to $r_{m+1}$. Since $V_1(0) \equiv V_2(0) \equiv \cdots \equiv V_m(0)$, for each $\ell$, $1 \leq \ell \leq m$, there exists an edge with label $((a,b),(c,d))$ from the root of $V_\ell(0)$ to a vertex corresponding to some robot $r_{i_\ell}$, where $r_{i_1} = r_{m+1}$. Now we show that the robots $r_{i_1}, r_{i_2}, \ldots, r_{i_m}$ are all distinct. To see this, note that by symmetry, there is an edge with label $((c,d),(a,b))$ from the root of $V_{i_\ell}(0)$, leading to a vertex that corresponds to robot $r_\ell$. So if $r_{i_1} = r_{i_2}$, for instance, then we have $r_1 = r_2$, a contradiction. Thus $r_{i_1}, r_{i_2}, \ldots, r_{i_m}$ are all distinct. Furthermore, since $V_1(0) \equiv V_2(0) \equiv \cdots \equiv V_m(0)$ and $V_{i_\ell}(0)$ is a subtree of $V_\ell(0)$ connected to the root of $V_\ell(0)$ by an edge with label $((a,b),(c,d))$ for each $\ell$, we have $V_{i_1}(0) \equiv V_{i_2}(0) \equiv \cdots \equiv V_{i_m}(0)$. Thus there are at least $m$ robots (including $r_{m+1}$) having a view equivalent to that of $r_{m+1}$. But then, there must be exactly $m$ such robots, since there cannot exist more than $m$ such robots by the definition of $m$. The lemma follows from this observation. □

**Lemma 3** *At time 0, the robots in the same group form a regular $m$-gon, and the regular $m$-gons formed by all the groups have a common center.*

**Proof** Suppose that $V_1(0) \equiv V_2(0) \equiv \cdots \equiv V_m(0)$, that is, $r_1, r_2, \ldots, r_m$ have an equivalent view at time 0. Consider the initial positions $p_1(0), p_2(0), \ldots, p_m(0)$ of these robots. Clearly at least one of $p_1(0), p_2(0), \ldots, p_m(0)$ is a corner of the convex hull $C$ of $\{p_1(0), p_2(0), \ldots, p_m(0)\}$. Then, since $V_1(0) \equiv V_2(0) \equiv \cdots \equiv V_m(0)$, each of $p_1(0), p_2(0), \ldots, p_m(0)$ must be a corner of $C$. Without loss of generality assume that $p_1(0), p_2(0), \ldots, p_m(0)$ occur in counterclockwise order around the convex hull. (See Figure 11.) Since $V_1(0) \equiv V_2(0) \equiv \cdots \equiv V_m(0)$, the internal angles of $C$ at the corners $p_1(0), p_2(0), \ldots, p_m(0)$ must be all identical, and the lengths of the

20

Figure 11: Illustration for the proof of Lemma 3, for the case $m = 4$.

edges of the convex hull must be all identical. (If $\overline{p_1(0)p_2(0)}$ looks shorter than $\overline{p_2(0)p_3(0)}$ to $r_2$, then $\overline{p_2(0)p_3(0)}$ should look shorter than $\overline{p_3(0)p_4(0)}$ to $r_3$, and so on, leading to a conclusion that $\overline{p_1(0)p_2(0)}$ is shorter than $\overline{p_1(0)p_2(0)}$, a contradiction.) So $p_1(0), p_2(0), \ldots, p_m(0)$ form a regular $m$-gon.

Suppose that at time 0, $r_{m+1}, r_{m+2}, \ldots, r_{2m}$ also have a view that are mutually equivalent, and that their respective positions $p_{m+1}(0), p_{m+2}(0), \ldots, p_{2m}(0)$ appear in counterclockwise order around the regular $m$-gon they form. Then again, since $V_1(0) \equiv V_2(0) \equiv \cdots \equiv V_m(0)$, the position of $p_{m+1}(0)$ relative to $p_1$ is the same as the position of $p_{m+2}(0)$ relative to $p_2$, and so on. (See Figure 11.) So the regular $m$-gon formed by $p_1(0), p_2(0), \ldots, p_m(0)$ and the regular $m$-gon formed by $p_{m+1}(0), p_{m+2}(0), \ldots, p_{2m}(0)$ have the same center. $\square$

**Lemma 4** *For any (deterministic) algorithm $\psi$, at any time instant $t$, the robots in the same group form a regular $m$-gon, and the regular $m$-gons formed by all the groups have a common center.*

**Proof** Suppose that $V_1(0) \equiv V_2(0) \equiv \cdots \equiv V_m(0)$, that is, $r_1, r_2, \ldots, r_m$ have an equivalent view at time 0. Now, since the initial distribution of the robots looks identical to $r_1, r_2, \ldots, r_m$, the new positions they compute using $\psi$ and their respective $Z_1, Z_2, \ldots, Z_m$ are all identical. Also, since $V_1(0) \equiv V_2(0) \equiv \cdots \equiv V_m(0)$, the center of the regular $m$-gon that $r_1, r_2, \ldots, r_m$ form at time 0 has the same $x$-$y$ coordinates in all of $Z_1, Z_2, \ldots, Z_m$. This means that $r_1, r_2, \ldots, r_m$ move in a symmetric manner relative to the center of the regular $m$-gon, and thus at time 1 they again form a regular $m$-gon with the same center. The same applies to all $n/m$ groups, and since the robots are synchronous, at time 1 they together form a collection of $n/m$ regular $m$-gons all having the same center. Since the robots in the same group have observed the same robot distributions, their next move at time 1 are also symmetric relative to the center of the regular $m$-gon they currently form. So again, at time 2 the robots form a collection of $n/m$ regular $m$-gons all having the same center. Continuing in the same manner, we can prove that at any time instant $t$, the robots form a collection of $n/m$ regular $m$-gons all having the same center. $\square$

Conversely, we have:

**Lemma 5** *For any multiset $F$ of points that can be partitioned into $n/m$ regular $m$-gons all having the same center, there exists a deterministic algorithm $\psi$ for forming a figure similar to $F$ starting from the initial configuration. (The algorithm does not depend on the initial configuration.)*

21

**Proof** We fix a total ordering over views. Also, we fix an ordering of the $n/m$ regular $m$-gons in $F$. The idea is to move the robots in the $j$-th group in the ordering of views to the corners of the $j$-th regular $m$-gon. Specifically, first the robots execute Algorithm $\psi_{getview}$, so each robot knows the rank of the group it belongs to. (Note that the robots can detect the termination of $\psi_{getview}$ simultaneously.) The robots in the first group need not move any more, since the $m$-gon they form are similar to the corners of the 1st $m$-gon of $F$ (except when the 1st $m$-gon is a point, in which case the robots must move to the point). Then each robot in the 2nd group computes the position of a corner of the 2nd $m$-gon of $F$ (relative to the location of the 1st $m$-gon of $F$) that is closest to its current position (breaking ties in any deterministic manner), and moves to that position. The robots in the 3rd group continues next, and so on. Then eventually, a figure similar to $F$ is formed. □

The following theorem summarizes the discussion given above.

**Theorem 11** *Let $m$ be the size of a largest subset of robots having an equivalent view at time 0. There exists a deterministic algorithm $\psi$ for forming a figure similar to a multiset $F$ of points iff $F$ can be partitioned into $n/m$ regular $m$-gons all having the same center.*

**Proof** The theorem follows from Lemmas 4 and 5. □

# 10  Concluding Remarks

We viewed a group of mobile robots as a distributed system in which the components can communicate with each other only by means of their moves, and investigated the possibility and impossibility of solving some of the problems related to the formation of geometric figures in the plane. Our study indicates that the assumptions we make on the knowledge and capabilities of the robots can affect the difficulty of solving the given problem in a subtle way. We are currently conducting similar investigations on (1) randomized algorithms, (2) the case in which the motion of a robot is not instantaneous, and (3) the 3-dimensional case. The results will be reported in a future paper.

# Appendix A

Under the assumption that clones do not exist, the following algorithm $\psi_{scatter}$ transforms any given distribution of any number of robots (in which some robots may occupy the same position) into one in which (1) no two robots occupy the same position, and (2) the robots are not located on a single line segment if $n > 2$. Part 1 of $\psi_{scatter}$ moves all the robots to distinct positions, and Part 2 assures that the robots do not form a single line segment. The idea in Part 1 is the following. Suppose $r_i$ and $r_j$ occupy the same position initially. We let each robot move repeatedly, in the

directions of their respective positive $x$-axes, over distances that depend on the unit distances of their respective local coordinate systems. Then since $r_i$ and $r_j$ are not clones, either they move in different directions, they move over different distances, or eventually only one of them becomes active and move. So $r_i$ and $r_j$ eventually distinct positions. We do this in such a way that no two robots that occupy distinct positions initially will move to the same position.

**Algorithm** $\psi_{scatter}$
**Part 1:** Suppose that robot $r_i$ becomes active and finds that not all robots occupy distinct positions. (Otherwise, Part 1 is over.) Let $P_r$ be the positions of the robots that $r_i$ observes, given in terms of $Z_r$.

**Case 1.1** If no other robot is located at the current position of $r_i$, then $r_i$ does not move, until it (becomes active later and) observes that all robots occupy distinct positions.

**Case 1.2** On the other hand, if $m$ other robots are located at the current position of $r_i$, where $m \geq 1$, then $r_i$ finds the distance $a_i > 0$ to its nearest neighbor (excluding those at the current position of $r_i$), and computes the value $b_i = a_i f(a_i)/2$, where for $x \geq 0$, $f(x) = 1 - 1/2^x$ is a monotonically increasing function with range $[0, 1)$. (Note that $a_i$ is given in terms of $Z_i$. So if another robot $r_j$ located at the same position as $r_i$ finds its own $a_j$ in terms of $Z_j$, then $f(a_i) \neq f(a_j)$ unless the unit distances of $Z_i$ and $Z_j$ are identical.) Then $r_i$ moves over distance $b_i/2$ in the positive $x$-direction of $Z_i$. After that, each time $r_i$ becomes active and finds that there are still $m$ other robots at the same position as itself, $r_i$ moves over distance $b_i/2^k$ in the same direction, if that is the $k$-th move ($k = 2, 3, \ldots$). When $r_i$ eventually becomes active and finds that there are fewer than $m$ other robots at the same position as itself, $r_i$ repeats this entire procedure from the beginning of **Case 1.2**, using a new value of $a_i$ (and resetting $k$). This process is repeated each time $r_i$ finds that fewer robots are located at the same position as itself, until no other robot is found to occupy the same position as $r_i$. Then $r_i$ waits, without moving, until all robots occupy distinct positions.

Then $r_i$ proceeds to Part 2.
**Part 2:** (At this moment, all the robots occupy distinct positions.) Suppose that robot $r_i$ becomes active and finds that the robots are located on a single line segment. (Otherwise Part 2 is over.) If $r_i$ is located at an endpoint of the segment, then it does not move. If $r_i$ is not at an endpoint, then $r_i$ moves over any distance in the direction perpendicular to the segment. (As soon as some robot does this, the robots no longer form a single line segment, and of course, all the robots occupy distinct positions.) □

**Lemma 6** *Algorithm $\psi_{scatter}$ transforms any given distribution of the robots into one in which (1) no two robots occupy the same position and (2) the robots are not located on a single line segment if $n > 2$.*

23

**Proof** In Part 1, no two robots that are at distinct positions ever occupy the same position, since during the execution of **Case 1.2** for each fixed value of $m$, robot $r_i$ moves over distance at most

$$b_i/2 + b_i/4 + \cdots < b_i = a_i f(a_i)/2 < a_i/2,$$

where $a_i$ is the distance from $r_i$ to any nearest robot not located at the same position at the beginning of **Case 1.2**. Since clones do not exist, any robot $r_j$ that initially occupies the same position as $r_i$ eventually moves to a different position than $r_i$, since either (a) their $x$-axes have different orientations, (b) their unit distances are different and thus $f(a_i) \neq f(a_j)$, or (c) only one of them becomes active and moves. In Part 2, since the robots at the endpoints of the line segment do not move, as soon as any one robot moves as specified, that robot and the robots at the endpoints form a triangle. $\square$

# References

[1] K. Fujimura, "Model of reactive planning for multiple mobile agents," *Proc. IEEE Int. Conf. on Robotics and Automation*, Sacramento, CA, June 1991, pp. 1503–1509.

[2] T. Fukuda and S. Nakagawa, "Approach to the dynamically reconfigurable robot systems," *Journal of Intelligent and Robotics Systems 1*, 1988, pp. 55-72.

[3] J. Y. Halpern and Y. Moses, "Knowledge and common knowledge in a distributed environment," *J. of the ACM 37*, 3, July 1990, pp. 549–587.

[4] N. Hirota, *private communication, 1992*.

[5] Y. Saito, *private communication, 1992*.

[6] M. Schneider, "Self–stabilization," *ACM Computing Surveys 25*, 1, 1993, pp. 45–67.

[7] K. Sugihara and I. Suzuki, "Distributed motion coordination of multiple mobile robots," *Proceedings of the 5th IEEE International Symposium on Intelligent Control*, Philadelphia, Pennsylvania, September 1990, pp. 138–143.

[8] O. Tanaka, *private communication, 1992*.

[9] O. Tanaka, M. Yamashita and I. Suzuki, "A note on motion coordination of distributed autonomous robots," (in Japanese) *Proceedings of the 1993 Joint Symposium on Electronics and Information*, Institute of Electronics, Information and Communication Engineers, Yamaguchi, Japan, October 1992.

[10] J. Wang and G. Beni, "Cellular robotic systems: Self-organizing robots and kinetic pattern generation," *Proc. of the 1988 IEEE International Workshop on Intelligent Robots and Systems*, Tokyo, Japan, 1988, pp. 139–144.

[11] Yamashita, M., and Kameda, T., "Computing on anonymous networks," *Proc. 7th ACM Symposium on Principles of Distributed Computing*, 1988, 117–130.
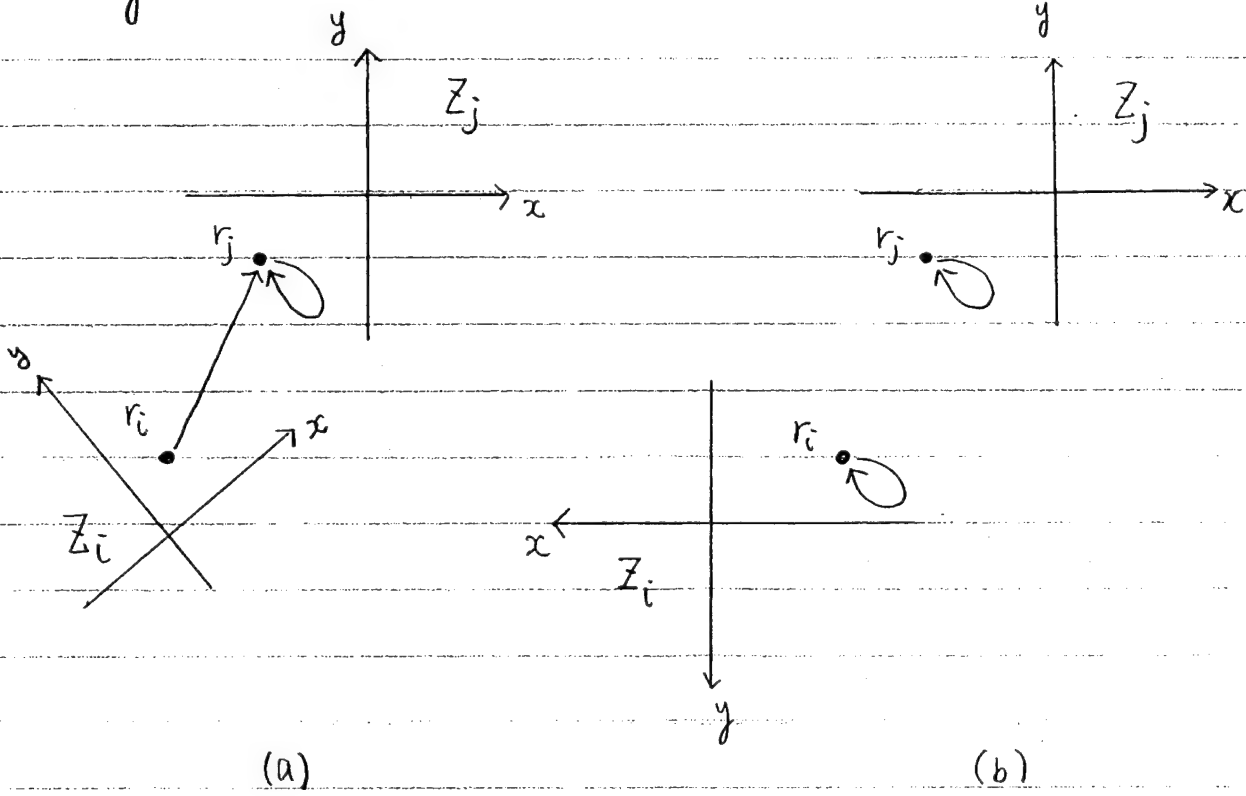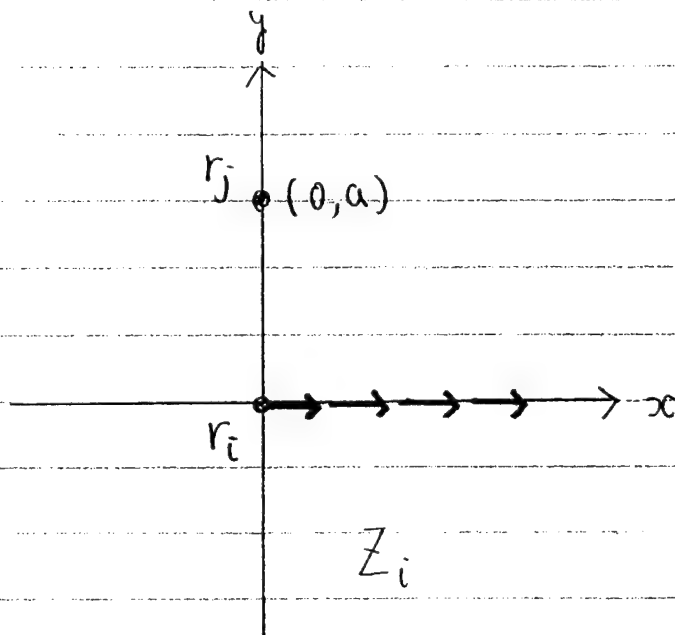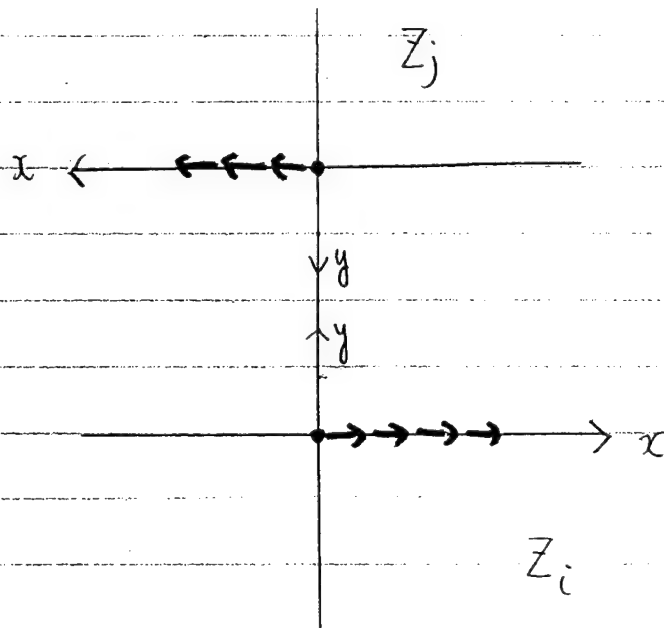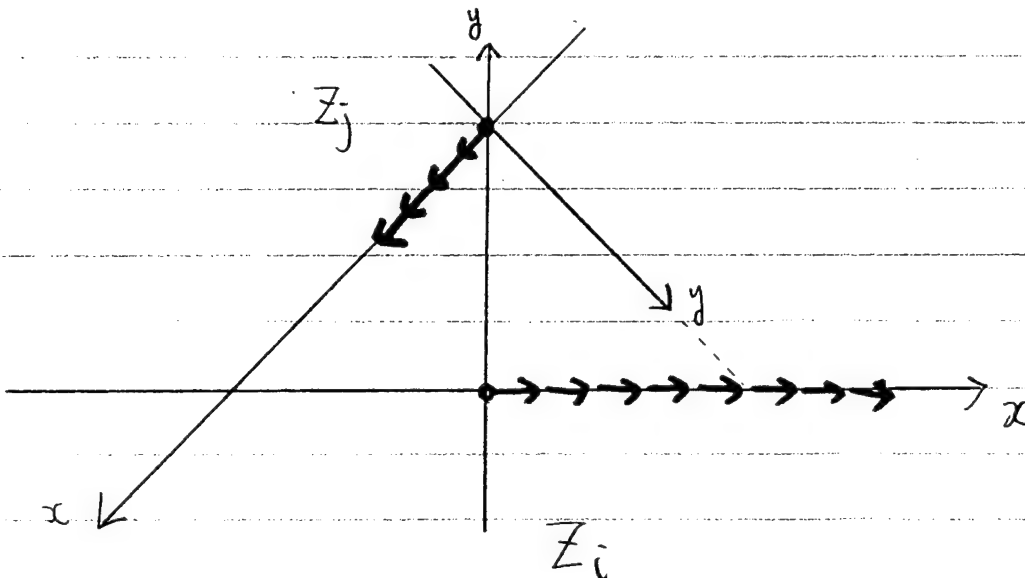
Fig. 2

(a)　　　　　(b)

Fig. 3

$y$

$r_j$ • $(0, a)$

$r_i$ → → → → → $x$

$Z_i$

Fig. 4

$Z_j$

$x$ ←←←←←•

↓$y$

↑$y$

•→→→→→→ $x$

$Z_i$

Fig. 5

Fig. 6
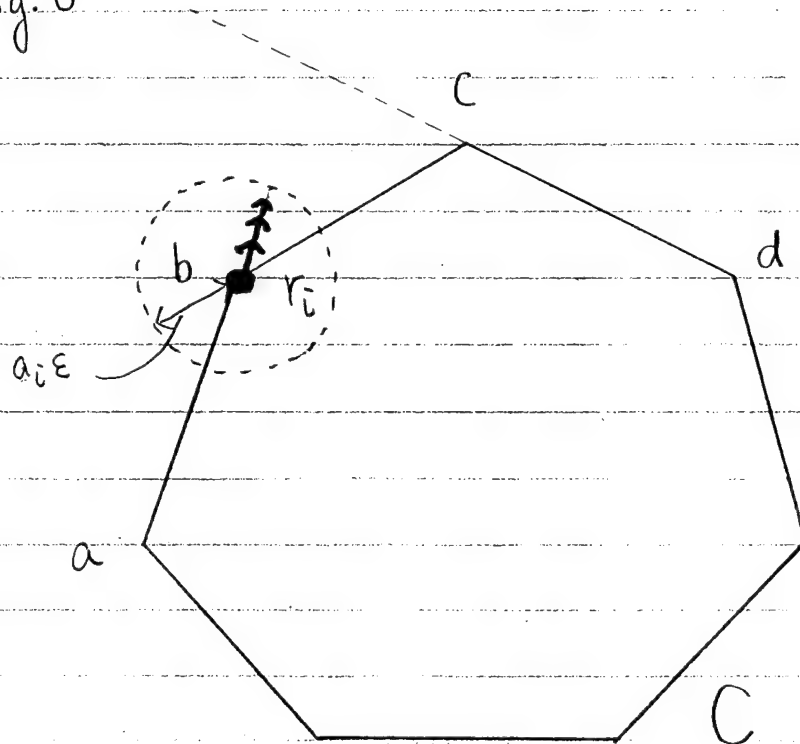
Fig. 7

$z_j$

$x \longleftarrow$

(1,1) of $z_i$

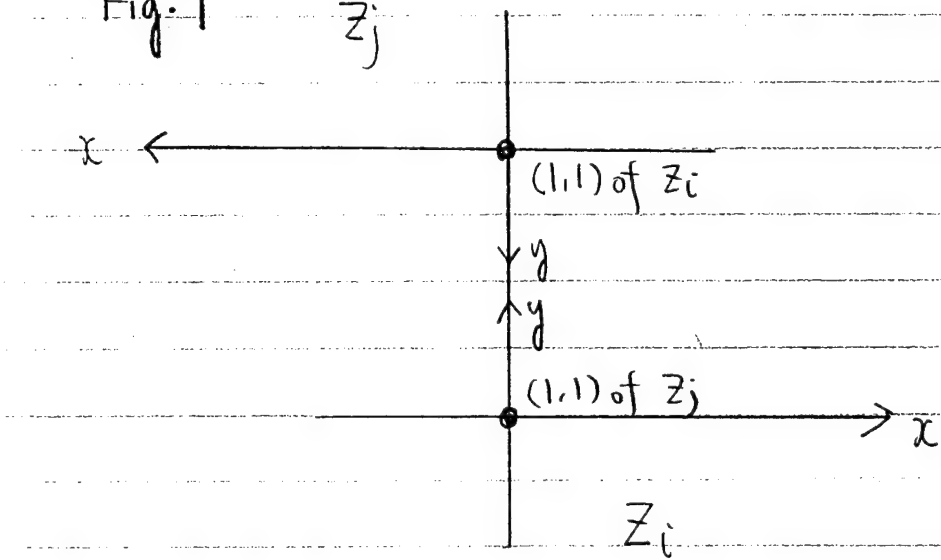$\downarrow y$
$\uparrow y$

(1,1) of $z_j$

$\longrightarrow x$

$z_i$

Fig. 8

unit distance

y-axis

x-axis

leader

Fig. 9

Fig. 10

Fig. 11